METHOD FOR QUERYING A DATABASE

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is based upon and claims priority to German Patent Application No. 10006959.2 filed February 16, 2000, the contents of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

The invention relates to a method for querying a database.

Such methods for querying databases are used to obtain information that falls within a certain search pattern. The databases store not only the data contents, such as measurement data, picture data, etc., but also characteristics, or so-called meta data, which characterize the data contents. In a database query, these characteristics, which satisfy the conditions of the query can thus be used, among other things, to classify the database content.

To be able to formulate such classification criteria, however, the physical memory structure of the characteristics and their semantic meaning has to be known.

Thus, the drawback is that corresponding requesting clients are always written for a specific database with a defined structure. If a user wants to direct an inquiry or query to a database, he must therefore know the corresponding client or query program or the data structure of the database.

In a query to a single database, a special client program for that database is therefore normally used. In case of a connection to the database via a network (LAN or WAN, Internet, etc.) the client program is loaded onto the client computer upon logging into the database, for example as a plug-in for a browser.

In contrast, if several databases are to be addressed simultaneously, this is realized via a corresponding interface, or middleware, of a server or a database. Several clients (or wrappers) are realized in the middleware on the server or in a database. The client is defined for the middleware. Since the wrappers are implemented in the middleware, the disadvantage is that only those databases that were previously analyzed with respect to their data structure and API (Application Program Interface) can be integrated via the middleware.

As a consequence, the known methods for querying databases depend on the latter's data structure to the extent that the respective structure of a database must be known for a query. This has the drawback that a query of a database is made more difficult, particularly if its exact structure is unknown to the inquirer.

SUMMARY OF THE INVENTION

Thus, the object of the invention is to create a method for querying databases, which permits a query or inquiry of a database whose exact structure is unknown.

This and other objects are attained by a method for querying a database. Referencing a standard structure according to the invention makes it possible to keep the query's structure general. This permits a simplified and general query since there is no longer any need to know the special structure of a database. In addition, one and the same query can be advantageously used for a plurality of databases with different special structures.

In an advantageous embodiment of the invention, both the query structure and the database structure are described by special descriptors, which reference standard descriptors. Since the standard descriptors themselves are available in addition, both a query and a database can be individually adapted for any application.

To compute the special descriptors from the standard descriptors and vice versa, a reference logic that is at least partially known in the database is used. It is also possible, however, to transmit said reference logic together with the query. This advantageously makes the structure of a query even more independent from the database structure, which can thus be kept more general.

In a preferred embodiment of the invention, upon a query, the matching standard and special descriptors contained in the database and the query are first determined. For non-matching descriptors, the associated descriptors are derived by means of a reference logic that is present or known in the database or is transmitted together with the query. For example, a standard descriptor of the database can be inferred from a special descriptor of the query, or a special descriptor of the database can be inferred from a standard descriptor of the query.

It is also feasible, however, to use reference logic to map a query's special descriptor to a standard descriptor which, in turn, is mapped to a special descriptor of the database via a corresponding reference logic, so that the query is even more independent of the database structure.

Since information on data contents can get lost in mapping or deriving a standard descriptor to a special descriptor and vice versa, it is also feasible, however, to convert a special descriptor of the query directly into a different but similar special descriptor of the database. The required computation logic can either exist in the database or be transmitted together with the query, as described above for the reference logic.

This makes it possible advantageously to avoid a potential loss of information.

To describe the special descriptors, semantic, physical and atomic elements can be used as the computation logic in order to define the semantic meaning, the physical memory structure and the link between memory structure and semantics. Although the individual atomic elements vary little in the different descriptors, in combination they advantageously result in a wide variation range.

If a database has a database structure that references a standard structure as described above, general queries that are largely independent from the database structure can advantageously be processed in said database.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention will now be described in greater detail, by way of example, with reference to the drawing in which:

Fig. 1 is a data scheme of a database of a clothing business to carry out the query method according to the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Fig. 1 shows a simple example of the method according to the invention referring to a fashion catalog of a clothing business A. As may be seen from this diagram, the descriptors "catalog number," "date," "color," "name," "material," "location" and "price" are defined as standard descriptors GD, typically in string format, and are known on both the query side and the database side.

On the database side, business A, in addition to using these standard descriptors GD, can also use special descriptors HD, such as "purchase date," "season," "description," "stock location," "retail store," and selling price," so that the structure of the database can be optimally adapted to the application.

These special descriptors HD in the database can be derived from the "standard descriptors" by a reference logic.

From the standard descriptor "date," for instance, both the special descriptor "purchase date" and the special descriptor "season" can be established. It should be noted that the only date known is usually the date of production or delivery, from which a purchase date and/or the corresponding season (summer, fall, spring, and winter collection) can at least approximately be established (e.g., by fixed typical time spans between production and purchase).

Conversely, the special descriptors "purchase date" and/or "season" can be used to infer at least a time span for a production date, which in Fig. 1 is indicated by the double arrows.

Thus, several, for example two, special descriptors HD may follow from one standard descriptor GD. As shown in Fig. 1, instead, it is also possible to derive only one special descriptor "selling price" from one standard descriptor "(purchase) price" by means of a corresponding reference logic, or one special descriptor "description" from several, for example two, standard descriptors "color" and "name," or vice versa.

The corresponding reference logic in this case may be, for example, a percentage markup (fixed profit margin) on the purchase price to determine the selling price. Another logic, however, may concern the position (prefix or postfix and separation by a hyphen) or the notation of the content of the special descriptor "description" (e.g., color in lower case letters and name in capital letters), so that said logic can be used to derive special descriptors from standard descriptors and vice versa.

To show the derivation possibilities in both directions in Fig. 1, the standard descriptors GD are linked with double arrows to special descriptors HD or in turn with standard descriptors GD.

The method for querying the database will now be described by means of the example of a special descriptor "description" with stored data content—for instance "red/blue pants"—and reference logic "name" in capital letters and "color" in lower case letters.

If a query is addressed to the database of business A, not only the employed special descriptors and standard descriptors must be defined as a query but also the standard descriptors

GD underlying the special descriptors HD. In the "description" example, this means, for instance, that all words in lower case letters in the value range of the query are interpreted as "color" and all words in capital letters as "name." Correspondingly, in case of a query with standard descriptors GD to the database of business A, the reference logic must be used to derive the special descriptors HD from the standard descriptors GD.

In contrast to the above example, however, the derivation from special descriptors HD to standard descriptors GD and vice versa may also be ambiguous, as described for the case of "purchase date," "season," and "date." Nevertheless, a query can achieve better selectivity with respect to both the "purchase date" and the "season" descriptor than with a general keyword search, or by even omitting the descriptor.

The proposed method thus allows queries to be made based on descriptors and their semantic meaning without any concrete knowledge of the data scheme.

In contrast to the described middleware solution of the prior art, the invention requires no central server, which knows the data schemes of the database, since each database itself is responsible for the derivation of the scheme from the standard descriptors. This does not imply, however, that changes must be made in the existing data scheme. This decentralized solution further makes it possible to integrate an unlimited number of databases in the search, where each database administrator is responsible for implementing the derivations and the matching criteria.

The scenario of a query of a database, which does not necessarily need to be a physical database, is for instance as follows, omitting trivial steps, such as evaluation of the matching standard descriptors GD:

1. The query is formulated on the client side with the available descriptors (standard and special descriptors) and is sent to the database.

2. The database first considers the special descriptors, forms the cut set with the special descriptors HD that are stored in the database, and evaluates it.

3. For the remaining special descriptors HD received, i.e., those that are not contained in the database, referencing standard descriptors GD are evaluated.

4. For the standard descriptors GD that are not used in the database, the database checks which of these standard descriptors were referenced by special descriptors HD and from the reference logic computes a query with these special descriptors HD.

The sequence of steps 1 to 4 is of course not mandatory for the method to function. A sequence of 1–4–2–3, for instance, is equally feasible. In addition, step 4, used advantageously to consult further descriptors for the query, is only optional, so that the query procedure according to the invention can be executed with only steps 2 and/or 3, depending on whether standard descriptors GD and/or special descriptors HD occur in the query.

This method makes it possible on the client side to use descriptors that are optimal for each application, without being limited to data, which was described by these special descriptors. It is further possible, prior to the database query, to determine the percentage share that a special descriptor has in the description of a data type (e.g., within the log-in protocol between client and server) and to load the reference logic (computation code and possibly GUI plug-in) of the most common descriptors in order to use the characteristics described by them efficiently. For this purpose it is feasible to store or reference in the database the computation code or GUI Applet of the special descriptors used in the database to compute the characteristics from the data that is to be described.

With the method according to the invention, descriptors deviating from the GDs can be used for description not only on the client side but also on the database side. This may allow for better indexing, for instance, without making impossible a query with the GD standard descriptors.

This example also shows that, compared with the standard descriptors GD, a special descriptor does not necessarily have to be a more selective descriptor.

It should be noted that to form an optimal cut set from the special descriptors in the database query and the special descriptors in the database, said descriptors should be defined as unambiguously as possible. If the computation code is realized with "distributed object" methods, the descriptors can be derived from the unambiguous IDs of the object classes (GUID/DCOM or IOR/CORBA). This has the advantage that the computation code is referenced together with the descriptor identification. A further possibility is to describe the descriptors by means of so-called atomic elements, which will be discussed below by way of example.

In queries with a plurality of descriptors, each with low information content (normally of the integer or string type), this method loses little information by derivations. But in complex descriptors with very specific information, which occur, for instance, in the description of pictures, the information loss is usually greater. Here, it would be advantageous to convert similar special descriptors HD directly without the large information loss due to derivation via standard descriptors GD.

We will now discuss, by means of an example, how special descriptors can be described in order to classify them as similar and to be able to convert one special descriptor HD into another, without knowing one of the two special descriptors.

For this purpose, special descriptors HD are described by three types of elements, so-called "atomic elements": semantic atomic element (sAE), physical atomic elements (pAE) and linking atomic elements (vAE), which define the <u>semantic</u> meaning, the physical <u>memory structure</u> of the stored quantity and the <u>link</u> between memory structure and semantics. This procedure results from the observation that the individual atomic elements in the different descriptors vary little but in combination produce a wide variation range. For example, histograms (vAE) are frequently used for the descriptors of images, but they vary in their granularity (pAE) and meaning (color, invariants, etc.) (sAE).

Corresponding to the procedure in the derivation via standard descriptors GD, the AEs, too, are standardized and known to both the client and the server. In contrast to the procedure in the standard descriptors GD, however, it may be necessary to update the AEs. This is particularly true for the sAEs, since here new descriptor types are more often introduced. In contrast, the pAEs and vAEs are less involved since they are subject to strong restrictions from the side of the database and the programming language.

Below, an example of the above-described procedure is discussed. It is assumed that the semantic AE (sAE) Color (), the physical AE (pAE) Integer, Real, Array<pAE> [ ] and the linking AE (vAE) Histogram () are declared as follows:

- Color (vAE, ColorSpace) represents a color description (sAE) with a vAE in the ColorSpace ∈ (RGB, YUV,...),
- Histogram (Integer Dim, (Real Bottom0, Real Top0, Real Step0),..., Array<pAE>[Integer (Top0-Bottom0)/Sept0],...) a multidimensional histogram.

A simple, three-dimensional color histogram 3DCHist in the RGB color space can thus be described as follows:

3DCHist: Color(Histogram(3, (0,255,32), (0,255,32), (0,255,32), Array<Array<Array<Integer>[ ]>[ ]>[ ]), RGB).

The histogram is thus three-dimensional and in each color space dimension (RGB) is divided into 8 bins.

A further color histogram 3DCHistLarge is described by

3DCHistLarge: Color(Histogram(3, (255,0,16), (255,0,16), (255,0,16), Array<Array<Array<Real>[ ]>[ ]>[ ]),RGB).

A query based on the histogram 3DCHistLarge can now be processed in a database realized with the descriptors in the 3DCHist format, without 3DCHistLarge having to be explicitly known in advance. For this purpose, the descriptor definition (similar to an interface definition) is transmitted suitably encoded. A conversion into the 3DCHist descriptor used in the database can then be effected based on this descriptor definition in that the sequence of the bins is exchanged (255,0 -> 0,255), the number of the bins per color space dimension is reduced, and their entries are suitably converted (Real -> Integer).

In contrast to a conversion or derivation of these mutually similar special descriptors on the client and database side via a standard descriptor, as described above, an unnecessary information loss can advantageously be avoided.

For example, the direct computation by means of the computation logic of a color histogram for the red component with 256 entries as a special descriptor on the client side into a color histogram for the red component with 128 entries as a special descriptor on the database side eliminates the unnecessary data loss that would occur if, as an intermediate step, a more remote (standard) descriptor were referenced, e.g., a color histogram for the red component with 8 entries.

This makes it possible to avoid an unnecessary information loss by selective transmission of the computation logic or the GUI for individual descriptors in addition to referencing general standard descriptors GD.

It should be noted that this special case of direct computation can of course be viewed as a query with a standard descriptor as opposed to an associated special descriptor on the database side, or vice versa. The reference logic is then represented by the computation logic or the computation rule.

The invention has been described in detail with particular reference to preferred embodiments thereof and examples, but it will be understood that variations and modifications can be effected within the spirit and scope of the invention.